CSCI 331

Generated by Doxygen 1.8.14

Contents

Chapter 1

Class Index

1.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Chapter 2

File Index

2.1 File List

Here is a list of all files with brief descriptions:

Chapter 3

Class Documentation

3.1 bplustreemk4 Class Reference

```
#include <bplustreemk4.h>
```

Public Member Functions

- bplustreemk4 ()
- bplustreemk4 (string directory)
- void processFile ()
- void cleanUp ()

3.1.1 Constructor & Destructor Documentation

3.1.1.1 bplustreemk4() [1/2]

```
bplustreemk4::bplustreemk4 ( )
```

```
                              {

}
```

3.1.1.2 bplustreemk4() [2/2]

```
bplustreemk4::bplustreemk4 (
               string directory )
```

```
                                {
//ofstream out;
fstream myfile;
this->dir = dir;
string header;
string headerout;
myfile.open(dir, ios::in);
myfile >> this->headerSize;
getline(myfile, header);
cout << header << endl;
this->header = header;
int fieldcount = 0;
int count = 0;
//a "different" way to read data in because it wasnt working
for (int i = 1; i < this->headerSize - 2; i++) {
    if (header.at(i) == '|') {
        cout << "encounter a field marker" << endl;
        fieldcount++;
        string temp;
        for (int x = i - count; x < i; x++) {
            temp.push_back(header.at(x));
        }
        if (fieldcount == 1) {
            this->insertFlag = stoi(temp);
        }
        if (fieldcount == 2) {
            this->totalNumRecords = stoi(temp);
        }
        if (fieldcount == 3) {
            this->sizeofRecords = stoi(temp)-1;
        }
        if (fieldcount == 4) {
            this->fieldsPerRecord = stoi(temp);
        }
        if (fieldcount == 5) {
            cout << "temp aka what ever we have deemed a field " << temp;
            /*vector<string> temp2;
            for (int y = 0; y < temp.size(); y++) {

            }*/
            vector<string> temp2;
            string buffer;
            stringstream ss(buffer);
            for (int y = 0; y < temp.size(); y++) {
                ss.clear();
                ss << temp.at(y);
                //cout << "Went into the string stream" << temp.at(y) << endl;
                ss >> buffer;
                //cout << "THIS IS WHAT IS IN THE BUFFER  " << buffer << endl;
                if (buffer != "\"" || buffer != "(" || buffer != ")") {
                    temp2.push_back(buffer);
                }
            }
            this->sizePerfield = temp2;
        }
        if (fieldcount == 6) {
            vector<char> temp3;
            for (int y = 0; y < temp.size(); y++) {
                if (temp.at(y) != '"'|| temp.at(y)!='(' || temp.at(y) != ')') {
                    temp3.push_back(temp.at(y));
                }

            }
            this->labelPerfield = temp3;
        }
        if (fieldcount == 7) {
            this->sortField = stoi(temp);
        }
        if (fieldcount == 8) {
            this->description = temp;
        }
        count = 0;
        i++;

    }
    count++;
}
myfile.close();
```

```
    this->rbnRoot = this->header.size();
    //debugging
    cout << this->headerSize << endl;
    cout << this->totalNumRecords << endl;
    cout << this->description << endl;

    /*for (int i = 0  ; i < this->sizePerfield.size(); i++) {
        cout << this->sizePerfield.at(i);

    }
    cout << endl;
    for (int i = 0; i < this->labelPerfield.size(); i++) {
        cout << this->labelPerfield.at(i);
    }*/
    //out.open("btree.dat");
    //out << header;
    //out.close();
    cleanUp();
    processFile();
}
```

3.1.2 Member Function Documentation

3.1.2.1 cleanUp()

```
void bplustreemk4::cleanUp ( )
```

```
                           {
    vector<string> tempf;
    string temp;
    for (int i = 0; i < this->sizePerfield.size(); i++) {
        if (this->sizePerfield.at(i) != "\"" && this->sizePerfield.at(i) != "(" && this->sizePerfield.at(i) != ")") {
            cout << this->sizePerfield.at(i);
            tempf.push_back(this->sizePerfield.at(i));
        }
    }
    this->sizePerfield = tempf;
    cout << endl;
    for (int i = 0; i < this->labelPerfield.size(); i++) {
        cout << this->labelPerfield.at(i);
    }
}
```

3.1.2.2 processFile()

```
void bplustreemk4::processFile ( )
```

```
                               {
    ofstream outfile;
    fstream myfile;
    myfile.open(this->dir);
    outfile.open("btree.dat");
    outfile << this->header;
    sequenceB sb;
    indexB ib;
    //pair temp;
    int nextP=0;
    int preP=0;
    this->totalSequnceBlocks = 0;
    string tempLine;
    while (getline(myfile, tempLine)) {
        string zip;
        if (tempLine.size() != this->sizeofRecords) {
            cout << "Line that was not big enough dected." << endl;
            continue;
        }
        else {
            cout << "The line is of valid lenghth " << endl;
            zip = tempLine.substr(0, 5);
            cout << " The zip is " << zip << endl;
            sb.nextPointer = this->headerSize + (this->blockSize * this->totalSequnceBlocks);
            cout << "Next pointer is " << sb.nextPointer << endl;
            cout << "pushing back this : " << tempLine << endl;
            sb.seqeunce.push_back(tempLine);
            cout << " SEQUENCE SIZE BEFORE ADDING " << sb.seqeunce.size() << endl;
            ib.seqeunce.push_back(kp(zip, (this->headerSize + this->totalSequnceBlocks * 249)));
            if (sb.seqeunce.size() == 3) {
                this->totalSequnceBlocks++;
                //cout << "this is what is in the sequence block " << sb.writeToFile() << endl;
                outfile << sb.writeToFile();
                sb.seqeunce.clear();
                continue;
            }
            if (ib.seqeunce.size() == this->indexBlockSize-1 ) {
                this->totalIndexBlock++;
                cout << "this is what is in the Index block " << ib.writeToFile() << endl;
                outfile << ib.writeToFile();
                ib.seqeunce.clear();
                continue;
            }
            //cout << "The sequence size after adding " << sb.seqeunce.size() << endl;
        }
    }
    outfile.close();
    myfile.close();
    cout << this->sizeofRecords << endl;
}
```

The documentation for this class was generated from the following files:

- bplustreemk4.h
- bplustreemk4.cpp

3.2 indexB Struct Reference

`#include <bplustreemk4.h>`

Public Member Functions

- indexB ()
- indexB (int prev, kp item)
- void setNext (int next)
- string writeToFile ()

Public Attributes

- int nextPointer
- int prevPointer
- vector< kp > seqeunce

3.2.1 Constructor & Destructor Documentation

3.2.1.1 indexB() [1/2]

```
indexB::indexB ( )   [inline]
```

```
         {

}
```

3.2.1.2 indexB() [2/2]

```
indexB::indexB (
              int prev,
              kp item )   [inline]
```

```
                          {
    this->prevPointer = prev;
    this->seqeunce.push_back(item);
}
```

3.2.2 Member Function Documentation

3.2.2.1 setNext()

```
void indexB::setNext (
              int next )   [inline]
```

```
                       {
    this->nextPointer = next;
}
```

3.2.2.2 writeToFile()

```
string indexB::writeToFile ( )   [inline]
```

```
                     {
    string temp;
    stringstream ss(temp);
    temp.push_back('i');
    temp.push_back('|');
    for (int i = 0; i < seqeunce.size(); i++) {
        ss >> temp;
    }
    temp.push_back('|');
    return temp;
}
```

3.2.3 Member Data Documentation

3.2.3.1 nextPointer

```
int indexB::nextPointer
```

3.2.3.2 prevPointer

```
int indexB::prevPointer
```

3.2.3.3 seqeunce

```
vector<kp> indexB::seqeunce
```

The documentation for this struct was generated from the following file:

- bplustreemk4.h

3.3 kp Struct Reference

```
#include <bplustreemk4.h>
```

Public Member Functions

- kp (string nkey, int value)
- void setValue (int nValue)
- int getValue ()

Public Attributes

- string key
- int value

3.3.1 Constructor & Destructor Documentation

3.3.1.1 kp()

```
kp::kp (
             string nkey,
             int value )   [inline]
```

```
                            {
         this->key = nkey;
         this->value = value;
     }
```

3.3.2 Member Function Documentation

3.3.2.1 getValue()

```
int kp::getValue ( )   [inline]
```

```
                   {
         return this->value;
     }
```

3.3.2.2 setValue()

```
void kp::setValue (
             int nValue )   [inline]
```

```
                            {
         this->value = nValue;
     }
```

3.3.3 Member Data Documentation

3.3.3.1 key

```
string kp::key
```

3.3.3.2 value

```
int kp::value
```

The documentation for this struct was generated from the following file:

- bplustreemk4.h

3.4 sequenceB Struct Reference

```
#include <bplustreemk4.h>
```

Public Member Functions

- sequenceB ()
- sequenceB (int prev, string item)
- void setNext (int next)
- string writeToFile ()

Public Attributes

- int nextPointer
- int prevPointer
- vector< string > seqeunce

3.4.1 Constructor & Destructor Documentation

3.4.1.1 sequenceB() [1/2]

```
sequenceB::sequenceB ( )  [inline]
```

```
            {

    }
```

3.4.1.2 sequenceB() [2/2]

```
sequenceB::sequenceB (
            int prev,
            string item )  [inline]
```

```
                            {
        this->prevPointer = prev;
        this->seqeunce.push_back(item);
    }
```

3.4.2 Member Function Documentation

3.4.2.1 setNext()

```
void sequenceB::setNext (
              int next )  [inline]
```

```
                          {
        this->nextPointer = next;
    }
```

3.4.2.2 writeToFile()

```
string sequenceB::writeToFile ( )  [inline]
```

```
                         {
        string temp;
        stringstream ss(temp);
        temp.push_back('s');
        //temp.push_back('|');
        //temp.push_back(prevPointer);
        temp.push_back('|');
        for (int i = 0; i < seqeunce.size(); i++) {
            for (int x = 0; x < seqeunce.at(i).size(); x++) {
                temp.push_back(seqeunce.at(i).at(x));
            }
            //ss << seqeunce.at(i);
            //cout << "What is going into the sequence " << seqeunce.at(i) << endl;
            //ss >> temp;
        }
        //cout << "THE LAST ITEM IN THE SEQEUENCE " << seqeunce.at(2)<<endl;
        //cout << "THE SIZE OF THE SEQUENCE BLOCK " << seqeunce.size() << endl;
        temp.push_back('|');
        temp.push_back(nextPointer);
        temp.push_back('|');
        //cout << "WHAT IS BEING RETURNED TO BE PRINTED " << temp << endl;
        //for (int i = 0; i < seqeunce.size(); i++) {
        //  cout << "IN THE SEQUENCE " << seqeunce.at(i) << endl;
        //}
        return temp;
    }
```

3.4.3 Member Data Documentation

3.4.3.1 nextPointer

```
int sequenceB::nextPointer
```

3.4.3.2 prevPointer

```
int sequenceB::prevPointer
```

3.4.3.3 seqeunce

```
vector<string> sequenceB::seqeunce
```

The documentation for this struct was generated from the following file:

- bplustreemk4.h

Chapter 4

File Documentation

4.1 bplustreemk4.cpp File Reference

```
#include <iostream>
#include <fstream>
#include <string>
#include <sstream>
#include <vector>
#include "bplustreemk4.h"
```

4.2 bplustreemk4.h File Reference

```
#include <string>
#include <sstream>
#include <vector>
```

Classes

- struct kp
- struct sequenceB
- struct indexB
- class bplustreemk4

4.3 main.cpp File Reference

```
#include <string>
#include <iostream>
#include <sstream>
#include <fstream>
#include "bplustreemk4.h"
```

Functions

- int main ()

Variables

- string dir = "C:\\Users\\reali\\Desktop\\CSCI\\us_postal_codesn+.txt"

4.3.1 Function Documentation

4.3.1.1 main()

```
int main ( )
```

```
           {
    /*Due to the lack of time this is implemented staticly*/
    bplustreemk4 test = bplustreemk4(dir);
    return 0;
}
```

4.3.2 Variable Documentation

4.3.2.1 dir

```
string dir = "C:\\Users\\reali\\Desktop\\CSCI\\us_postal_codesn+.txt"
```

Index